CROEQS: Contemporaneous Role Ontology-based Expanded Query Search — Implementation and Evaluation

Stijn Vandamme Johannes Deleu Tim Wauters Brecht Vermeulen Filip De Turck
INTEC Broadband Communication Networks
Ghent University — IBBT
Ghent, Belgium
{Stijn.Vandamme, Johannes.Deleu, Tim.Wauters, Brecht.Vermeulen, Filip.DeTurck}@intec.ugent.be

***Abstract*—Searching annotated items in multimedia databases becomes increasingly important. The traditional approach is to build a search engine based on textual metadata. However, in manually annotated multimedia databases, the conceptual level of what is searched for might differ from the high-levelness of the annotations of the items. To address this problem, we present CROEQS, a semantically enhanced search engine. It allows the user to query the annotated persons not only on their name, but also on their roles at the time the multimedia item was broadcast. We also present the ontology used to expand such queries: it allows us to semantically represent the domain knowledge on people fulfilling a role during a temporal interval in general, and politicians holding a political office specifically. The evaluation results show that query expansion using data retrieved from an ontology considerably filters the result set, although there is a performance penalty.**

Keywords—semantic search; ontologies; databases; performance

I. INTRODUCTION

There is still a large gap between the information that can automatically be extracted from the multimedia data and the interpretation that the same data has for the users in different situations. This gap is called *the semantic gap*. To overcome this gap, and facilitate search and retrieval, items in multimedia databases are generally manually annotated with textual metadata.

Our work was performed on a database that contains broadcasted television material from Flemish public and commercial broadcasters. At the moment of writing, the database contains about 5000 hours of broadcasted television material, for the most part (3 500 hours) news. That corresponds to about 70000 individually annotated news items.

Each item is manually annotated with both keywords from a dictionary and a "free text" description. 63000 different keywords are tagged, and 139000 different words used in the description. Additional metadata for each multimedia item includes the programme title, the date on which the item was broadcasted, etc. In total, the textual metadata comprises 440 MiB.

We created first a traditional search engine, a system that indexes all this textual metadata and uses these indexes to respond to (textual) queries.

Annotators can however never foresee all the possible scenarios in which the user would want to find and retrieve a given item. There can be a gap between the conceptual level of how items are annotated and the high-levelness of what the user is searching for.

As a use case, we chose political news. Searching for multimedia items in this domain, users don't always know which politician has spoken out on a certain subject: they may want to express queries as "show me items about (previous) prime ministers co-occurring with a given political subject" or "show me items about a politician from a given party in connection with a given political subject".

To address this problem, we designed and implemented an enhanced search engine, built on top of the textual search engine.

Essential in our approach here is the separation of *the annotations about the video fragments*, and *the domain knowledge about the subject matter* (in our case, Belgian politics).

In this paper, we first describe how we designed and implemented a traditional search engine (section II), and a semantically enhanced one (section III) on top of the first. For the latter, we created an ontology with the relevant domain knowledge, which we describe in section IV. In section V, we evaluate the trade-off between the performance and the achieved gains in terms of result set refinement.

II. TRADITIONAL SEARCH ENGINE

We used Lucene [1] to build a traditional search engine. The engine indexes (after stemming) the textual metadata.

We also built a user interface to expose this search capacity, in the form of a web application based on the Jetty web server [2].

When the user enters a query in the textbox and hits "Search", the query is sent to the search engine using a HTTP POST request. The HTTP response includes the total number of items matching the query, and the details of the first ten results.

The results are sorted: those with a high relevance score are listed first. The relevance is determined by the sum of the squares of the relevance weights of the different queried words (or phrases) for the result item. The more frequently the word occurs in the result item, the higher the relevance

weight. The higher the percentage of items annotated with that same word, the lower the weight for that term [3].

In the Lucene query syntax, a "+" before a clause means that the item's text must match the clause, while two clauses just separated with whitespace are interpreted as a disjunctive query, although the relevance will be higher when both clauses are matched.

The textual search engine is deployed on a host with two dual-core AMD Opteron processors 2212 and 4 GiB RAM.

III. SEMANTICALLY ENHANCED SEARCH ENGINE

Our enhanced system has a very similar user interface as the traditional Lucene-based search engine: the user can enter a query in the textbox, hit search, generating a HTTP POST request. The response contains the number of result items and the item details for the first 10 results, sorted by relevance score.

The difference lies in the expressivity of the queries. Our system extends the query possibilities with semantic clauses.

In our system, a semantic clause is a condition that can be matched by politicians, either during a certain interval or during their whole political career. Our system provides 3 kinds of semantic clauses: "a politician belonging to party *x*", "a politician responsible for competence *y*", and "a politician with role *z*", where *z* can be Minister, Secretary of State, etc.

A. System description

Figure 1 shows how the enhanced search engine is set up: when our system receives a query with a semantic clause, the query is translated into a standard query, which is sent to the traditional search engine.

The semantic clause is translated into a list of politicians that match the semantic clause, with (optionally) the time interval in which the politician fulfilled the condition.

To store the domain knowledge to make the translation, we created an ontology about Belgian politics. We describe it in detail in section 4. For the implementation and internal use of our ontology, we use Jena [4], a Semantic Web toolkit. The ontology data itself is stored in a MySQL triple database. For the query translation, our politics ontology is queried using SPARQL [5].

This query translation service is deployed on an AMD 2200 MHz processor with 1 GiB RAM.

B. Query translation

Basically, the semantic clause is replaced by a long disjunctive ("or") expression of all the politicians which did match the clause and the interval in which they matched it.

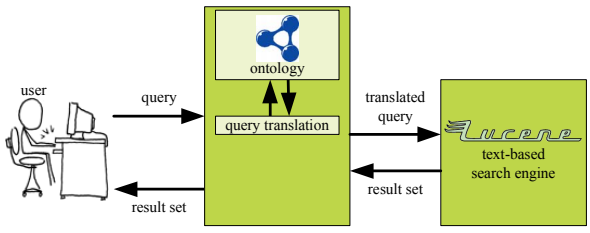

Figure 1. CROEQS: schematic overview

In the translated query, that time interval is used in a broadcast date clause. A possible answer will be included in the result set if it is tagged with a person having held the requested role, and broadcasted during the time interval in which that person held that role. We use the broadcast date here as an approximation of the "subject date" of the video item. This approximation is usually adequate, but it is not perfect: sometimes a video item describes events of the (recent or not-so-recent) past, in which different people might have held the requested role.

IV. CONTEMPORANEOUS ROLE ONTOLOGY

In computer science, an ontology (also referred to as a domain model) is a representation of a common conceptualization of a particular domain. It provides a shared understanding of a domain, and can be communicated across people and application systems.

Typically, an ontology contains a hierarchical description of important concepts in a domain, and describes crucial properties of each concept. An ontology may additionally contain relations between these concepts. Classes, properties and relations constitute the ontology's model. Once the model is defined, an ontology can contain individuals belonging to these classes, having specific values for the properties, and being related to other individuals through the defined relations.

In RDF [6], all information is stored in a single universal data structure: the triple. A triple consists of three resources: a subject, a predicate and an object. The ontology model is defined using the RDF Schema [7] and OWL [8] vocabularies, both W3C recommendations for the Semantic Web.

We created with Protégé [9] a core ontology about people fulfilling roles during certain time intervals, and then extended it for the specific domain of (Belgian) politics.

A. Core ontology model

In the core ontology (figure 2, shadowed part), we want to be able to express that a person holds or has held a specific function, which often is associated with a particular organization, during a given interval. Additionally, the name of the organization can change over time.

For example, since October 1, 2005, prof. dr. Paul Van Cauwenberge fulfills the role of "Rector" at "Universiteit Gent", a university that was named "Rijksuniversiteit Gent" before 29 June 1991.

Naturally, a person can hold multiple roles (even in multiple organizations) at the same time. Roles can also be held by multiple people during the same or overlapping intervals. Typically, the "membership" role in different organizations is held by quite a lot of people simultaneously.

Person: A natural person, with a name.

Role: A role, a function, a mandate, a position or an office, often associated to a particular organization, which can be held or fulfilled by persons, usually with duties, trusts, honors or responsibilities attached to it. Typically, a role is held ("fulfilled") by one or more persons at the same time, who are succeeded by others as time progresses.

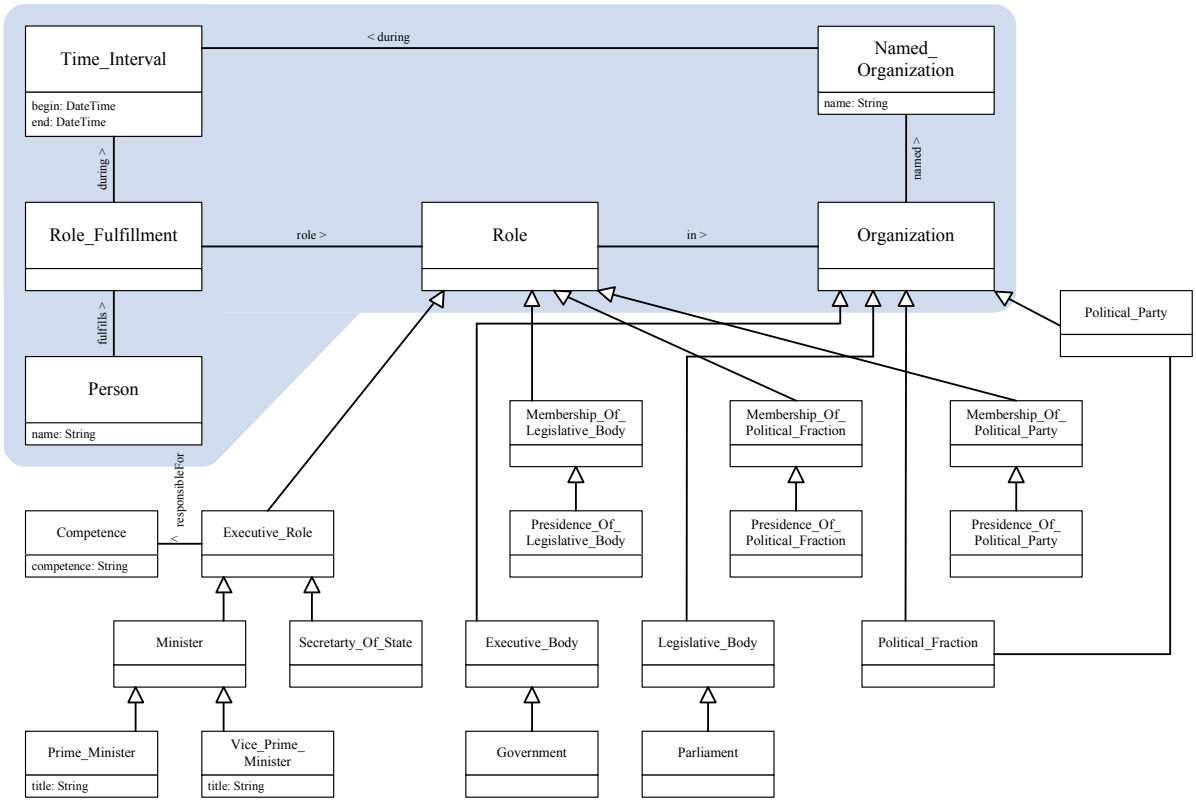

Figure 2. Role and Politics Ontology
The shadowed part is generic, the rest is the extension for the domain of politics

Organization: Roles are often associated with a particular organization, club, party or institution.

Named Organization: The name of an organization is a property that can change over time. Since this ontology models temporal domain knowledge, we could not simply add "name" as a basic property of the Organization class. Instead, the organization is related to a named organization, which has a "name" property which was the organization's name during a temporal interval.

Time Interval: A temporal interval, from a given start date to a given end date. Both dates are optional: if the start date is empty, the interval is considered to have started before time immemorial. If the end date is empty, the interval hasn't ended yet (until further notice). The end date cannot proceed the start date.

Role Fulfillment: This class connects the dots: a role fulfillment is one specific person fulfilling one specific role during a specific time interval.

B. *Extended ontology: politics*

Political parties, executive and legislative political bodies are all subclasses of Organization (figure 2).

The Belgian political system is rather complex: Belgium is a federal state simultaneously divided in Communities and Regions; consequently there are multiple executive and legislative bodies.

An additional concept in the politics extension is the class Competence. An executive role may be related to one or more competences through the relationship "responsible_for". For example, a "minister of Justice and Institutional Reform" (the role is an individual resource belonging to the class Minister) is responsible for the competences "Justice" and "Institutional Reform".

We populated our ontology with all federal ministers and secretaries of state since 1978, all Flemish ministers since 1981, and a number of influential politicians of all Belgian political parties over the last decades. In total, the ontology contains 464 politicians.

V. EVALUATION

We evaluate the system based on 20 queries which combine each one hot topic in Belgian politics (expressed in a single word) with one semantic clause.

An example of such a query is "+responsible:Education +strike". Both plus signs indicate that the result set should only include items annotated with both the subject "strike" and a politician's name responsible for education in a period contemporaneous with the broadcast.

A. *Refinement versus performance trade-off*

We evaluate the result set size returned by our enhanced system, compared to the query where we leave out the semantic clause and only keep the subject.

In the example, we compare the original query to the single keyword query "+strike".

The result set of the query including the semantic clause is always a subset of the result set of the query in which the semantic clause is dropped, as the semantic clause (starting with a +) acts as a filter.

The most important performance metric is the query response time. High and unpredictable response times decrease user satisfaction.

Figure 3 shows the trade-off between this restriction of the result set size and the response time.

Since we compare the query with semantic clause to the single keyword query, the relative result set size of the single keyword is 1 by definition. The response time is quasi-constant for all single keyword queries: 110 milliseconds on average.

Adding the semantic clause reduces the result set by a factor 0.25 to 0.0005. The average is 0.067, the standard deviation 0.087. This is especially useful when performing a targeted search (as opposed to browsing and surfing through search results without clear end goal).

On the other hand, the response time is several times larger: between 245 and 1826 milliseconds. The average is 824 ms, the standard deviation 444 ms.

B. A deeper look at the response time

To understand why the response time varies considerably between different queries with different semantic clauses, we take a deeper look.

The response time is the sum of four components: the time necessary to translate the query (by our system), the time necessary to parse the translated query (by the Lucene-based engine), the time necessary to retrieve the metadata (by the Lucene-based engine), and the network time.

In the network time, we only include the time in which the translated query and the response crosses over the network between het hosts that run the traditional and the enhanced search engine. We do not measure the time between any user's machine and the web application. Both search engine hosts, and the test machine from which we measure are connected to a 100 Mbps LAN-network.

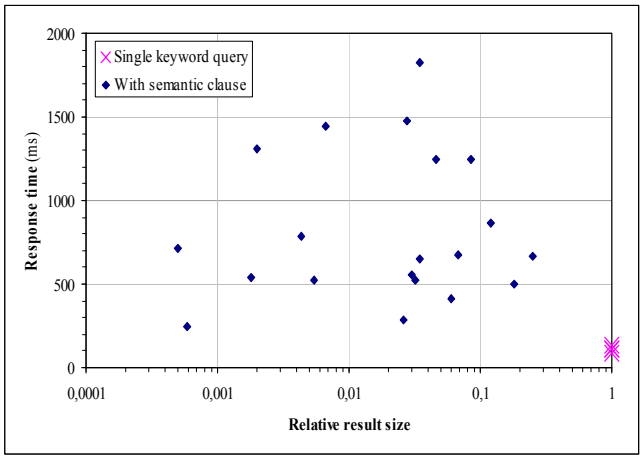

Figure 3. Refinement versus performance trade-off
Comparing CROEQS with single keyword query

The network time can fluctuate quite a bit between two particular executions of a query, but averaged out over several executions of the same query, we find it is consistently 70 milliseconds, independent of the query.

The other three components do depend from query to query. Figure 4 shows that these components are highly correlated to the number of politician-interval combinations that match the semantic clause. The more combinations match the semantic clause, the longer it takes to translate the query, the longer it takes to parse the translated query at Lucene-side and the longer it takes to evaluate and retrieve the result set.

The correlations are linear: the correlation coefficients are 93.3 % for the translation time, 97.9 % for the parsing time, and 96.7 % for the retrieval time.

As a consequence, the result time becomes unacceptably long when the amount of politicians matching the semantic query becomes too high. This is for instance the case when you query on the members of a large political party. If we consider that a response time of more than 1.5 seconds is unacceptable, the semantic query should not have more than 73 matches.

VI. FUTURE WORK

Our core ontology can be extended into other specific domains, and even more general ontologies, even in the context of enhancing multimedia search in news items.

There is of course plenty of other information, besides the roles people fulfill throughout their careers, that can be used to semantically improve multimedia searching.

A deeper focus on performance may improve speed, especially in the cases where a large amount of concepts matches the semantic query. Possible approaches include further parallelization and evaluating the benefit of the sorting of the result set that takes place at Lucene-side.

VII. RELATED WORK

Multimedia search based on textual descriptions is already in use: on Internet scale, Google offers Google Video where surrounding text on Internet pages is used as text to be indexed. Blinkx [10] offers audio and video search based on text transcripts obtained using speech recognition. Lienhart [11] presented video search based on text automatically optically recognized in the video itself, both scene text and artificially added text, such as name titles, tickers, etc.

Search engines with special syntax to restrict the result set size based on technical metadata, are also already in use. For instance, Google Video users can ask only to get results in a given file type or only from a given internet domain. The Lucene search engine library allows system designers to define their own keywords, allowing users to restrict the query results to only those items for which a predefined field has a certain value (or lies in a certain value range).

Popescu et al. [12] use WordNet [13] for enhancing the retrieval of images about animals, extending a queried concept with its WordNet subconcepts. In the given example, a query for "dog" would also include images

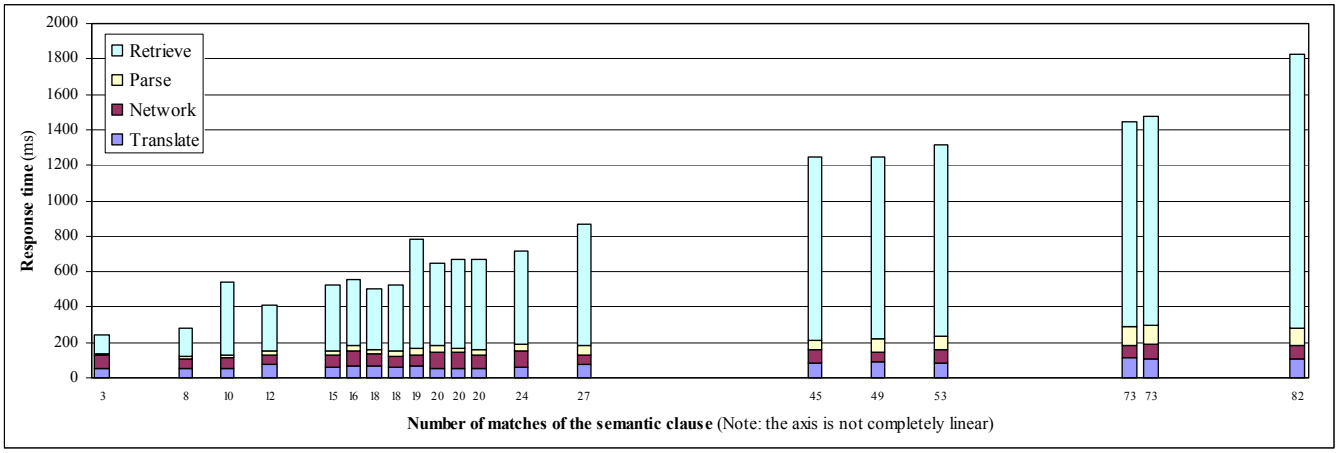

Figure 4. Query response time as a function of the number of matches of the semantic clause

matching "pug", "papillon", "collie" and "beagle". Via the user interface, this concept hierarchy can be browsed, and the results are updated accordingly.

For the same problem of image retrieval in the domain of the animal kingdom, Wang et al. [14] present a multi-modality ontology-based approach, where the high-level ontology with animal concepts is linked to a medium level textual ontology containing information about habitats and distribution and a low-level visual ontology with concepts like color and texture.

Smith et al. [15] are working on standardizing a more general large-scale concept ontology for multimedia.

The alternative approach to video searching is based on extracting information from the multimedia content itself: text and speech extraction fall in this category; also techniques like camera shot segmentation, key frame extraction and high-level feature extraction are heavily researched and often combined. [16]

VIII. CONCLUSIONS

We presented an ontology for expressing that people fulfill a role during a temporal interval, and we presented a search engine where users can use this information to restrict their result set.

Our results show that the approach of translating queries based on ontologies allows for more expressive and targeted queries. The semantic clause restricts the result set size considerably, which is helpful when doing targeted search. A quantitative study provides a limit to the amount information that can be found in the ontology and subsequently used in the expanded query, before performance becomes an issue.

REFERENCES

[1] E. Hatcher and O. Gospodnetic, "Lucene in Action", Manning Publications, 2005

[2] Jetty, `http://www.mortbay.org/jetty/`, 2 September 2008

[3] Documentation for org.apache.lucene.search.Similarity, `http://lucene.apache.org/java/2_2_0/api/org/apache/lucene/search/Similarity.html`, August 2007

[4] J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne and K. Wilkinson, "Jena: Implementing the Semantic Web Recommendations", Proc. of the 13th int'l World Wide Web conf. on alternate track papers & posters, ACM, 2004, pp. 74–83

[5] SPARQL Query Language for RDF, `http://www.w3.org/TR/rdf-sparql-query/`, 15 January 2008

[6] RDF Primer, `http://www.w3.org/TR/rdf-primer/`, 10 February 2004

[7] RDF Vocabulary Description Language 1.0: RDF Schema, `http://www.w3.org/TR/rdf-schema/`, 10 February 2004

[8] OWL Web Ontology Language – Overview, `http://www.w3.org/TR/owl-features/`, 10 February 2004

[9] N. Noy, M. Sintek, S. Decker, M. Crubézy, R. Fergerson and M. Musen, "Creating Semantic Web Contents with Protégé-2000", IEEE Intelligent Syst., 16:2, 2001, pp. 60–71

[10] B. Alfonsi, "Blinkx – Ups the Ante in the Search and RSS Markets," IEEE Distributed Syst. Online, 6:8, 2005

[11] R. Lienhart, "Automatic text recognition for video indexing", Proc. of the 4th ACM int'l conf. on Multimedia, ACM, 1997, pp. 11–20

[12] A. Popescu, P.-A. Moëllic and C. Millet, "SemRetriev – An ontology driven image retrieval system", Proc. of the 6th ACM int'l conf. on image and video retrieval, ACM, 2007, pp. 113–116.

[13] G. Miller, "WordNet – A lexical database for English", Communications of the ACM, 38:11, 1995, pp. 39–41

[14] H. Wang, L.-T. Chia, and S. Liu, "Image retrieval ++ – Web image retrieval with an enhanced multi-modality ontology", Multimedia Tools and Applications, Springer, 39:2, 2008, pp. 1380–7501

[15] J. R. Smith, M. Naphade, J. Tesic, C. Shih-Fu, W. Hsu, L. Kennedy, A. Hauptmann and J. Curtis, "Large-scale concept ontology for multimedia", IEEE Multimedia, 13:3, 2006, pp. 86–91

[16] Y. Li, S.-H. Lee, C.-H. Yeh and C.-C. J. Kuo "Techniques for movie content analysis and skimming", IEEE Signal Processing Mag., 23:2, 2006, pp. 79–89